Key elements of the servlet response are,

1. A status code.(for whether the request was successful)
2. Content-type (text, picture etc.)
3. The Content (the actual HTML, image etc)

There are eight methods in HTTP 1.0,

GET, POST, HEAD, TRACE, DELETE, CONNECT, OPTIONS, PUT.

**Difference b/w GET and POST method:**

1. Post request has the body but the Get request does not have the body.
2. In Post request the parameters are send as a body so they are not limited but the Get request appends the parameters in the URL.
3. Security concern.
4. Bookmark.
5. Get is generally used for getting things (just like a select) from server but Post is meant to use for sending data to be processed (just like update).
6. Get is an idempotent and Post is not an idempotent. Idempotent means same request can sends twice on server with no negative consequences.
7. Post is not a DEFAULT method. If you do not put method=”POST” in your form, the default is an HTTP GET method.

**set/ add response headers:**

setHeader() overwrites the existing value.

addHeader() adds an additional value.

If no value present in the header then both works in the same way.

response.setHeader(“foo”,”bar”);

response.addHeader(“foo”,”bar”);

response.setIntHeader(“foo”,1234);

response.setContentType(“text/html”) is same as below method,

response.setHeader(“content-type”, “text/html”);

**Redirect vs. Request Dispatch:**

**Redirect**

* a redirect is a two step process, where the web application instructs the browser to fetch a second URL, which differs from the original
* a browser reload of the second URL will not repeat the original request, but will rather fetch the second URL
* redirect is marginally slower than a forward, since it requires two browser requests, not one
* objects placed in the original request scope are not available to the second request

how to redirect a request,

response.sendRedirect(“http://www.google.com”);

Instead of writing the full URL you can also provide the part of the URL as below.

If the original client URL is

<http://www.wickedlysmart.com/myApp/cool/bar.do>

the servlet calls sendRedirect() with a relative URL that does NOT start with a forward slash:

sendRedirect(“foo/stuff.html”);

<http://www.wickedlysmart.com/myApp/cool/foo/stuff.html>

sendRedirect(“/foo/stuff.html”);

<http://www.wickedlysmart.com/foo/stuff.html>

**Forward/ Dispatch**

* a forward is performed internally by the servlet
* the browser is completely unaware that it has taken place, so its original URL remains intact
* any browser reload of the resulting page will simple repeat the original request, with the original URL

How to dispatch a request,

RequestDispatcher view = request.getRequestDispatcher(“result.jsp”);

view.forward(request,response);

**Interface SingleThreadModel**

Deprecated. As of Java Servlet API 2.4, with no direct replacement.

public interface SingleThreadModel

Ensures that servlets handle only one request at a time. This interface has no methods.

If a servlet implements this interface, you are guaranteed that no two threads will execute concurrently in the servlet's service method. The servlet container can make this guarantee by synchronizing access to a single instance of the servlet, or by maintaining a pool of servlet instances and dispatching each new request to a free servlet.

Note that SingleThreadModel does not solve all thread safety issues. For example, session attributes and static variables can still be accessed by multiple requests on multiple threads at the same time, even when SingleThreadModel servlets are used. It is recommended that a developer take other means to resolve those issues instead of implementing this interface, such as avoiding the usage of an instance variable or synchronizing the block of the code accessing those resources.